In [3]:

```python
import numpy as np
X = np.array(([2,9],[1,5],[3,6]),dtype=float)
y = np.array(([92],[86],[89]),dtype=float)
X = X/np.amax(X,axis=0)
y=y/100
def sigmoid(x):
    return 1/(1+np.exp(-x))
def derivatives_sigmoid(x):
    return x*(1-x)
epoch = 1000
learning_rate = 0.6
inputlayer_neurons = 2
hiddenlayer_neurons = 3
output_neurons = 1
wh = np.random.uniform(size=(inputlayer_neurons,hiddenlayer_neurons))
bh = np.random.uniform(size=(1,hiddenlayer_neurons))
wo = np.random.uniform(size=(hiddenlayer_neurons,output_neurons))
bo = np.random.uniform(size=(1,output_neurons))
for i in range(epoch):
    net_h = np.dot(X,wh) + bh
    sigma_h = sigmoid(net_h)
    net_o = np.dot(sigma_h,wo)+bo
    output = sigmoid(net_o)
    deltaK = (y-output)*derivatives_sigmoid(output)
    deltaH = deltaK.dot(wo.T) * derivatives_sigmoid(output)
    wo = wo+sigma_h.T.dot(deltaK) * learning_rate
    wh = wh+X.T.dot(deltaH)*learning_rate

print(f"Input:\n{X}")
print(f"Actual output:\n{y}")
print(f"Predicted Output:\n{output}")
```

Run Code

```
for i in range(epoch):
    net_h = np.dot(X,wh) + bh
    sigma_h = sigmoid(net_h)
    net_o = np.dot(sigma_h,wo)+bo
    output = sigmoid(net_o)
    deltaK = (y-output)*derivatives_sigmoid(output)
    deltaH = deltaK.dot(wo.T) * derivatives_sigmoid(output)
    wo = wo+sigma_h.T.dot(deltaK) * learning_rate
    wh = wh+X.T.dot(deltaH)*learning_rate

print(f"Input:\n{X}")
print(f"Actual output:\n{y}")
print(f"Predicted Output:\n{output}")
```

```
Input:
[[0.66666667 1.        ]
 [0.33333333 0.55555556]
 [1.         0.66666667]]
Actual output:
[[0.92]
 [0.86]
 [0.89]]
Predicted Output:
[[0.89461257]
 [0.88163521]
 [0.89420665]]
```

In [ ]: